From Heuristics to Higher Structures: A Categorical and Homotopical Formalization of Autonomous Architecture AI Discovery

Douglas C. Youvan

doug@youvan.com

July 26, 2025

In July 2025, the ASI-Arch system was introduced as a breakthrough in artificial intelligence—a multi-agent framework capable of autonomously discovering high-performing neural network architectures with minimal human guidance. Described as an "AlphaGo moment for model architecture discovery," the system explored nearly two thousand models, producing over a hundred state-of-the-art designs. However, while its empirical success is notable, the underlying mathematics remains opaque. ASI-Arch operates as a heuristic-driven search over architectures, lacking formal definitions of the space it traverses, the transformations it applies, and the novelty it uncovers. This paper proposes a mathematically rigorous alternative. We reformulate autonomous architecture discovery using the language of Category Theory and Homotopy Type Theory (HoTT). Architectures are treated as objects in a category; transformations as morphisms; and functional equivalence as homotopies. Discovery is modeled functorially, with novelty corresponding to the identification of distinct homotopy classes. Scientific progress emerges as the structured traversal of a higher-dimensional space of possibilities. Our approach not only clarifies what discovery means in this context—it builds a scalable, interpretable, and theoretically grounded framework for the future of autonomous scientific research.

Keywords: ASI-Arch, autonomous discovery, category theory, homotopy type theory, neural architecture search, functor, scientific epistemology, topos theory, emergent intelligence, AI research agents, model equivalence, structure-preserving transformation, machine creativity, formal methods in AI, higher category theory, HoTT, innovation scaling, scientific formalism, computational epistemology. A collaboration with GPT-4o. CC4.0.

I. Introduction

In the ongoing evolution of artificial intelligence, model architecture design has traditionally remained a domain of human creativity, guided by intuition, trial-and-error, and incremental innovation. The recent emergence of ASI-Arch marks a provocative shift in this paradigm. Introduced in mid-2025, ASI-Arch is presented as the first fully autonomous system capable of performing model architecture discovery without human intervention. Its creators describe it as an "AlphaGo moment" for architecture research—a reference to the iconic 2016 Go match where artificial intelligence revealed strategic possibilities far beyond human conception. Over the course of 1,773 computational experiments and 20,000 GPU hours, ASI-Arch produced 106 novel architectures that surpassed the state of the art in linear attention models.

However, this breakthrough is largely empirical. The inner mechanics of ASI-Arch—its modular agents (Planner, Coder, Trainer, Analyst), its search strategies, and its emergent design principles—remain conceptually opaque. The system is described in heuristic and black-box terms. No formal structure is provided to explain how discovery occurs, how novelty is defined, or how equivalence between models is tracked.

This paper seeks to go deeper. Our goal is to recast architecture discovery as a mathematically principled process. By adopting the tools of Category Theory and Homotopy Type Theory (HoTT), we define the space of architectures as a structured category, where models are objects and architecture-altering transformations are morphisms. We treat discovery not as random sampling, but as structured exploration—guided by functorial mappings, topological distinctions, and compositional consistency. Novelty, under this view, becomes a homotopical event: the identification of a distinct equivalence class within a landscape of structured possibilities.

By formalizing autonomous discovery in this way, we aim to replace black-box heuristics with transparent epistemology, paving the way for a deeper understanding of how machines can not only learn, but theorize, create, and understand.

II. Summary of ASI-Arch

1. System Architecture: Modular Agents in a Closed Scientific Loop

ASI-Arch is structured around a modular, multi-agent system that mimics the core stages of the scientific process: hypothesis generation, implementation, experimentation, and analysis. The four primary components are:

- Planner: Selects which architectural hypotheses to test. It operates over a combinatorial space of model configurations, driven by prior evaluations and a loosely defined novelty heuristic.
- Coder: Translates planner selections into executable architecture definitions—likely via a domain-specific language or dynamic code generation pipeline.
- Trainer: Instantiates, trains, and evaluates the architecture using standardized datasets and performance metrics, generating scalar results for downstream feedback.
- Analyst: Synthesizes results across trials, updating the planner's priorities and estimating the promise of unexplored regions of the architecture space.

These agents operate in a closed-loop configuration, with asynchronous communication and no human-in-the-loop feedback once initialized. This creates the appearance of autonomous scientific behavior.

---

2. Empirical Results and Emergent Behaviors

The system ran 1,773 architecture trials, consuming 20,000 GPU hours, and produced 106 linear attention models that outperformed human-designed state-of-the-art (SOTA) architectures on benchmark tasks. Notably, these models were not refinements of known paradigms but displayed unexpected structural features, suggesting that the system had explored regions of the architecture space unfamiliar to human researchers.

The authors compared this to AlphaGo's famous Move 37, in which the AI defied conventional wisdom but ultimately revealed a superior strategy. ASI-Arch's discoveries were characterized as "emergent"—not engineered, but arising spontaneously from the autonomous dynamics of the system.

---

3. Claimed Scaling Law of Discovery

A key assertion in the paper is that scientific discovery itself is subject to a scaling law—analogous to those observed in large language models and other deep learning systems. While precise formulations are not disclosed, the implication is that:

$$\text{Discovery Quality} \propto (\text{Compute})^{\alpha}$$

This scaling behavior is said to govern not only performance (e.g., accuracy or loss) but also qualitative novelty—i.e., the emergence of structural innovations that would not be found under human guidance or constrained search heuristics.

If valid, this would represent a profound shift: scientific advancement itself becomes a function of compute rather than cognition.

---

4. Limits of the Current Formulation

Despite the novelty of its results, the ASI-Arch framework suffers from several critical limitations:

- Heuristic opacity: The planning strategy is not specified mathematically. There are no objective functions or policy update rules disclosed.
- Black-box evaluation: Architectures are treated as monolithic entities; no effort is made to define a structure-preserving representation or graph-theoretic analysis.

- No formal space of architectures: The system explores an ill-defined search space with unknown topological or categorical structure. There is no notion of metric distances, deformation, or equivalence classes.
- Undefined novelty: The term “novel” is used colloquially but lacks grounding in information theory, topology, or any formal metric.

These gaps make ASI-Arch powerful as an empirical phenomenon, but conceptually shallow. It succeeds without understanding—a phenomenon we aim to correct by establishing a principled mathematical foundation for autonomous discovery.

III. Formalizing the Architecture Space as a Category

To move beyond heuristic search, we propose to model the space of neural architectures as a mathematical category. This categorical structure provides a rigorous language for reasoning about compositionality, transformability, and equivalence within architecture design. We denote this category as:

$$\mathcal{C}_{\text{Arch}} : \text{The Category of Neural Architectures}$$

---

1. Objects: Neural Architectures

The objects of $\mathcal{C}_{\text{Arch}}$ are complete, parameterizable neural network architectures. Each object $A \in \text{Ob}(\mathcal{C}_{\text{Arch}})$ represents:

- A directed acyclic graph (DAG) of computational modules (layers, operations)
- A topological structure with defined dataflow
- An instantiation blueprint for training and evaluation

Mathematically, we may think of each architecture as a **functor** from a syntactic DAG schema D_A to a category of tensor operations:

$$A : D_A \to \mathcal{C}_{\text{TensorOps}}$$

This captures both the architecture’s graph structure and the semantics of computation.

2. Morphisms: Compositional Transformations

Morphisms in $\mathcal{C}_{\text{Arch}}$ are structure-preserving **transformations between architectures**. Given two architectures A and B, a morphism $f : A \to B$ corresponds to a finite sequence of compositional edits, such as:

- Insertion or deletion of a layer
- Modification of a connection pattern (e.g., residual shortcut)
- Substitution of operations (e.g., replacing ReLU with GELU)
- Parameterizing a module family (e.g., kernel size generalization)

We constrain morphisms to be semantically valid: they must preserve data compatibility (e.g., tensor shapes) and maintain DAG-acyclicity.

Each morphism induces a commutative diagram between the dataflows of A and B, ensuring that structural transformations maintain coherent information propagation.

---

3. Identity Morphisms and Composition Rules

For every architecture A, there exists an **identity morphism** $\text{id}_A : A \to A$, representing the trivial transformation.

Morphisms compose: if $f : A \to B$ and $g : B \to C$, then $g \circ f : A \to C$ is the morphism obtained by applying f then g. Composition is associative and respects identity:

$$\text{id}_B \circ f = f = f \circ \text{id}_A$$

This gives $\mathcal{C}_{\text{Arch}}$ the structure of a **small category**—provided the number of architectures under consideration is finite or indexed.

---

4. Subcategories: Structured Model Families

Specific architecture classes can be modeled as **full subcategories** of $\mathcal{C}_{\text{Arch}}$. For example:

- $\mathcal{C}_{\text{CNN}} \subseteq \mathcal{C}_{\text{Arch}}$: convolutional architectures
- $\mathcal{C}_{\text{Transformer}} \subseteq \mathcal{C}_{\text{Arch}}$: self-attention-based models
- $\mathcal{C}_{\text{LinearAttn}} \subseteq \mathcal{C}_{\text{Transformer}}$: architectures with linear attention mechanisms

Each subcategory inherits morphisms from $\mathcal{C}_{\text{Arch}}$, constrained to remain within the architectural paradigm defined by the subcategory. This structure permits targeted exploration within a defined family while preserving the general compositional framework.

---

5. Toward a Category of Equivalence Classes

Later in this paper, we refine $\mathcal{C}_{\text{Arch}}$ into a homotopical setting, where morphisms can be interpreted up to equivalence (e.g., functionally identical models despite superficial differences). This leads naturally to the definition of a quotient category or ∞-groupoid, where discovery corresponds to movement across connected components of model space.

Before that, however, we must formalize how evaluations and novelty enter this categorical landscape—which we approach via functorial mappings in the next section.

IV. Discovery as Functorial Action

In classical science, the act of discovery involves mapping structural forms to observable effects—laws, patterns, or metrics. In our categorical framework, we model this mapping as a functor between the category of architectures and the category of evaluations:

$$F : \mathcal{C}_{\text{Arch}} \longrightarrow \mathcal{C}_{\text{Eval}}$$

This functor translates architectural form into evaluative substance, preserving the compositional structure of transformations between models and their measurable

consequences. Scientific discovery, in this formalism, is the emergence of previously unobserved mappings under this functor—i.e., lifting new structural configurations into the evaluative space.

---

1. The Evaluation Category $\mathcal{C}_{\text{Eval}}$

We define $\mathcal{C}_{\text{Eval}}$ as a category where:

- **Objects** are evaluation profiles, i.e., tuples such as:

$$E = (\text{Accuracy}, \text{FLOPs}, \text{Training Time}, \text{Novelty Score}, \text{Robustness})$$

- **Morphisms** are interpretable refinements or degradations between evaluation states. For instance:
  - $\phi : E_1 \to E_2$ could represent a drop in accuracy due to architectural pruning.
  - These transformations are **monotonic** or **partial order-preserving** in nature.

The morphisms in $\mathcal{C}_{\text{Eval}}$ reflect empirical transitions: they capture the **semantic consequences** of design changes encoded in $\mathcal{C}_{\text{Arch}}$.

---

2. Definition of the Functor *F*

The functor F must satisfy:

- **Object Mapping**: For each architecture $A \in \text{Ob}(\mathcal{C}_{\text{Arch}})$, $F(A) = E \in \text{Ob}(\mathcal{C}_{\text{Eval}})$ is the evaluation result after training and testing A.
- **Morphism Mapping**: For each transformation $f : A \to B$ in $\mathcal{C}_{\text{Arch}}$, the functor maps it to an evaluation transition $F(f) : F(A) \to F(B)$, reflecting how the change in architecture modifies observable outcomes.

This defines F as a **structure-preserving map**:

$$F(\text{id}_A) = \text{id}_{F(A)}, \quad F(g \circ f) = F(g) \circ F(f)$$

Hence, **compositional changes in architectures result in compositional changes in evaluation metrics**.

---

3. Discovery as Lifting across Functors

The key insight is that **scientific discovery** corresponds to **lifting a new object or morphism into the codomain of F**:

- A **new architecture** $A' \in \mathcal{C}_{\text{Arch}}$ is a discovery *only* when $F(A')$ yields an evaluation profile **not isomorphic** to those already mapped:

$$\nexists A \text{ such that } F(A) \cong F(A')$$

- A **new transformation** $f : A \to B$ is a discovery if $F(f)$ reveals a **non-trivial causal or performance relationship** between existing evaluations.

Thus, **discovery = functorial emergence** of previously unmapped or non-equivalent evaluative states from structural variation.

---

4. Composition Preserves Interpretability and Constraints

One of the central challenges in neural architecture search is the loss of interpretability due to arbitrary or adversarial transformations. By insisting on functorial structure, we ensure that:

The key insight is that **scientific discovery** corresponds to **lifting a new object or morphism into the codomain of F**:

- A **new architecture** $A' \in \mathcal{C}_{\text{Arch}}$ is a discovery *only* when $F(A')$ yields an evaluation profile **not isomorphic** to those already mapped:

$$\nexists A \text{ such that } F(A) \cong F(A')$$

- A **new transformation** $f : A \to B$ is a discovery if $F(f)$ reveals a **non-trivial causal or performance relationship** between existing evaluations.

Thus, **discovery = functorial emergence** of previously unmapped or non-equivalent evaluative states from structural variation.

This contrasts with black-box search, where transformations are often ad hoc and their effects must be rediscovered from scratch each time.

---

5. Beyond Performance: Functorial Mapping to Discovery Metrics

The codomain category $\mathcal{C}_{\text{Eval}}$ may be extended beyond raw performance metrics to include meta-evaluations:

- Novelty Scores (e.g., distance in architecture embedding space)
- Surprise (e.g., deviation from predicted outcomes)
- Explainability (e.g., activation trace complexity)
- Theoretical Insight (e.g., interpretable motifs)

These too become functorial targets, allowing us to reason about what kind of discovery has occurred and not just whether performance improved.

In this view, discovery becomes not a stochastic accident, but a mappable, traceable transition in a structured mathematical landscape, respecting both interpretability and compositional design. The next step is to formalize novelty and equivalence in this landscape—an endeavor we approach homotopically.

V. Novelty and Equivalence: A Homotopical View

The concept of *novelty* in architecture discovery cannot be fully captured by scalar metrics like accuracy or parameter count alone. To understand whether a discovered model is truly "new" in a structural or functional sense, we must formalize equivalence—and more deeply, the *space* of transformations that connect different models. We turn to homotopy theory to do this, interpreting neural architectures not just as discrete objects, but as points in a connected space of structural transformations. This shift allows us to treat discovery as the emergence of homotopically distinct classes—regions of the architecture space not reachable by continuous deformation from existing models.

---

1. Homotopy Between Models

In classical topology, a *homotopy* is a continuous deformation between two functions or spaces. Analogously, we define a **homotopy between neural architectures** $A \simeq B$ as a **sequence of semantically-preserving transformations** that map A into B, such that:

- Intermediate models maintain functional behavior
- Evaluative profiles remain invariant or equivalent under the functor F

Symbolically, a homotopy is a higher-order morphism:

$$H : A \Rightarrow B$$

where H is not a single morphism $f : A \to B$, but a **path** f_t through morphism space, with $f_0 = \mathrm{id}_A$, $f_1 = f$, and all intermediate architectures along f_t being functionally equivalent under F.

---

2. Path Spaces: Continuous Deformation Between Architectures

We define the **path space** $\mathrm{Path}(A, B)$ as the set of all homotopies from A to B. These paths form a topological space or an ∞-groupoid:

- If $\mathrm{Path}(A, B) \neq \emptyset$, then A and B are **connected** via functional deformation.
- If $\mathrm{Path}(A, B) = \emptyset$, then A and B are in **distinct homotopy classes**—functionally and structurally novel with respect to one another.

The total space of architectures thus has the shape of a **homotopical landscape**, where novelty corresponds to **moving between disconnected components** or **encountering higher-dimensional holes** (e.g., loops or cycles in architecture evolution).

---

3. Discovery as Identification of Distinct Homotopy Classes

With this structure, we can now define **scientific discovery** as the **identification of a new homotopy class** in $\mathcal{C}_{\mathrm{Arch}}$:

$$\text{Novel Architecture } A' \in \mathcal{C}_{\mathrm{Arch}} \text{ such that } \nexists A \text{ with } A' \simeq A$$

In this framing:

- Minor variations within a known class are seen as deformations, not discoveries.

- Radically distinct models, which cannot be reached via smooth transformations, are genuinely new.

This framework provides a mathematical underpinning to the colloquial idea of "emergence"—the appearance of a solution type not reachable from prior knowledge by known steps.

---

4. Link to Homotopy Type Theory (HoTT)

Homotopy Type Theory provides a powerful interpretation of these ideas. In HoTT:

- Types are spaces, and terms are points in those spaces.
- Paths between terms correspond to proofs of equality.
- Higher paths represent proofs of equality between proofs, creating an infinite tower of structure.

We interpret architectures as types, and model identity as equivalence under functional behavior:

$$A = B \quad \text{iff} \quad A \simeq B$$

This is a form of the **univalence axiom**:

> "Equivalent models are indistinguishable in all meaningful contexts."

In HoTT, this means:

$$\mathrm{Id}_{\mathcal{C}_{\mathrm{Arch}}}(A, B) \simeq \mathrm{Equiv}_{\mathcal{C}_{\mathrm{Eval}}}(F(A), F(B))$$

This gives us a robust formal criterion for saying when two models are "the same"—not just syntactically, but behaviorally and evaluatively.

---

5. Practical Implications for ASI Systems

With this homotopical view, we can endow autonomous architecture search systems with:

- Novelty detectors based on transitions between homotopy classes
- Equivalence checkers that avoid redundant evaluations
- Path-based exploration heuristics that follow deformation flows
- Univalence-aware compression: storing only representative models per class

This reframing transforms black-box NAS into a structured exploration of a topologically rich model space, guided not just by metrics, but by a deep understanding of how models are related.

In the next section, we extend this structure into the higher-dimensional space of agent behavior and recursive learning loops—framing discovery as traversal within a higher category.

VI. Search as Higher-Order Structure Traversal

The previous sections treated architecture discovery as the traversal of a structured categorical and homotopical space. We now turn our attention to the agents conducting this traversal—specifically, the autonomous modules within systems like ASI-Arch (Planner, Coder, Trainer, Analyst). While previously described in heuristic terms, these agents can be rigorously modeled as higher-level functors, each mediating between a task space and the architectural landscape. Their interaction forms a higher category, with discovery emerging as coherent movement through a network of transformations, refinements, and evaluations.

---

1. Autonomous Agents as Higher-Level Functors

Each agent Π_i within the discovery system performs a conceptual function: mapping from **tasks**, goals, or performance constraints into specific architectural decisions. We define:

$$\Pi_i : \mathcal{C}_{\text{Task}} \to \mathcal{C}_{\text{Arch}}$$

Where:

- $\mathcal{C}_{\text{Task}}$: The category of problem specifications, including objective functions, datasets, constraints, and historical context.
- $\mathcal{C}_{\text{Arch}}$: The previously defined category of neural architectures.

In this framework:

- The Planner is a functor selecting candidate architectures based on search heuristics or learned priors.
- The Coder refines high-level plans into lower-level instantiable blueprints, serving as a *refinement functor*.
- The Trainer acts as a dependent functor, whose behavior varies over a fibered category of runtime conditions (e.g., compute resources).
- The Analyst maps the history of evaluations into adjustments in the other functors' behavior—an endofunctor on the system's learning dynamics.

Each of these functors can themselves be objects in a 2-category, where morphisms are natural transformations between functors—e.g., adapting Planner behavior as Analyst feedback accumulates.

---

2. Feedback Loops as a 2-Category or ∞-Category

The full structure of autonomous discovery is not a flat composition of actions, but rather a recursive, feedback-rich hierarchy of mappings. These layered interactions are best modeled not in a 1-category, but in a 2-category, where:

- **Objects**: Categories like $\mathcal{C}_{\text{Task}}$, $\mathcal{C}_{\text{Arch}}$, and $\mathcal{C}_{\text{Eval}}$
- **1-Morphisms**: Functors between these categories (e.g., Π_i)
- **2-Morphisms**: Natural transformations between functors—e.g., learned adaptations of agents based on past outcomes

In more complex systems, we may even require an ∞-category structure to account for:

- Higher-order corrections (updates to the updates)
- Learning-to-learn strategies (meta-learning)
- Reflexive models where agents model each other

This opens the door to modeling autonomous scientific agents not as pipelines, but as functorial networks of evolving intent, bound together by coherence conditions.

---

3. Discovery Paths as Higher-Dimensional Diagrams

In a higher category, commuting diagrams encode the consistency of multi-stage processes. We define a discovery path as a higher-dimensional diagram—not just a linear sequence of transformations, but a structured traversal through tasks, architectures, and evaluations, obeying compositional constraints.

- A commuting square might represent:
    - Two alternative architecture paths leading to the same evaluation
    - Functional equivalence via two distinct model-building strategies
- A commuting cube might represent:
    - A consistent reconfiguration across all agents (Planner, Coder, Trainer) given a task shift
    - Structural stability across multiple feedback loops

These diagrams serve not merely as notational devices, but as semantic invariants of scientific coherence. The very notion of "reproducibility" in autonomous discovery may be defined as the commutativity of such diagrams across reruns, datasets, or tasks.

---

4. Operationalizing Higher Traversal in Autonomous Systems

Modeling discovery as higher-order traversal provides concrete benefits:

- Meta-learning becomes morphism-level adaptation: changes in functors themselves
- Inter-agent communication becomes structure-preserving transformation: coherent exchange between functorial agents
- Emergent strategies can be understood as limit constructions—the outcome of converging diagrams
- Failure modes (e.g., hallucinated novelty) correspond to broken diagrams or non-commuting transformations

The structure not only explains what these agents do, but how they do it consistently, and how to prove correctness, convergence, or contradiction in their behavior.

In summary, viewing autonomous discovery through the lens of higher category theory allows us to encode recursive feedback, agent adaptation, and system-wide consistency as mathematical invariants of traversal. This perspective shifts the paradigm from heuristic NAS systems to coherent mathematical explorers—machines that reason, revise, and reveal, all within a unified and principled framework. In the next section, we generalize this even further by situating the architecture space within a topos, enabling a global epistemology of AI-led science.

VII. Toward a Topos of Scientific Discovery

To complete our mathematical formalization of autonomous architecture discovery, we now take a global perspective. So far, we've modeled architectures as objects in a category, their transformations as morphisms, their evaluations as functorial images, and their search dynamics as higher-order traversals. But science is not merely a series of local experiments—it is the synthesis of scattered observations into coherent global understanding.

To model this process, we invoke the tools of topos theory. Specifically, we treat the architecture space as a site, define sheaves to represent measurable outcomes, and interpret scientific discovery as the gluing of local knowledge into global theories. The result is a flexible, powerful epistemological framework in which AI systems like ASI-Arch can be understood not just as search engines, but as knowledge-producing entities operating within a logical universe.

---

1. Defining a Site: Architecture Space with Grothendieck Topology

Let us define a **site** $(\mathcal{C}_{\text{Arch}}, J)$, where:

- $\mathcal{C}_{\text{Arch}}$ is the category of neural architectures (as developed earlier).
- J is a **Grothendieck topology** on this category, specifying **coverings** of each object.

In this setting:

- A covering of an architecture A is a family of morphisms $\{f_i : A_i \to A\}$ such that the A_i collectively capture all relevant "local views" of A.
- For example, $\{A_i\}$ could represent:
    - Pruned subgraphs
    - Low-resolution approximations
    - Parameter subsets
    - Equivalent model variants under functional deformation

This topology encodes how architectural information is distributed and reconstructed across variations.

2. Sheaves Represent Observables: Performance, Surprise, Cost

Given the site $(\mathcal{C}_{\text{Arch}}, J)$, we define a **sheaf** $\mathcal{F}$ as a contravariant functor:

$$\mathcal{F} : \mathcal{C}_{\text{Arch}}^{\text{op}} \to \mathbf{Set}$$

Each architecture A is mapped to a set $\mathcal{F}(A)$ of **observable quantities**, such as:

- $\mathcal{F}_{\text{Perf}}(A)$: Performance metrics (e.g., accuracy, generalization gap)
- $\mathcal{F}_{\text{Cost}}(A)$: Resource usage (e.g., compute time, memory footprint)
- $\mathcal{F}_{\text{Surprise}}(A)$: Degree of deviation from prior expectations

These sheaves assign to each architecture its **measurable outcomes**, with **restriction maps** ensuring that observables are **consistent under architectural simplification or transformation**:

$$\mathcal{F}(A) \xrightarrow{f^*} \mathcal{F}(A') \quad \text{for } f : A' \to A$$

Intuitively, these maps let us pull back observations from complex models to their substructures—enabling **local analysis**.

---

3. Discovery as Gluing Local Observations into Global Theories

The central idea of sheaf theory is **gluing**: if we observe a family of local models $\{A_i\}$ that cover a global model A, and their associated observables are **locally compatible**, then we can reconstruct a **unique global observable** on A.

This captures the **epistemology of scientific discovery**:

- **Experimentation** corresponds to collecting local observables $\mathcal{F}(A_i)$.
- **Analysis** checks for compatibility across overlaps $A_i \cap A_j$.
- **Synthesis** glues these local pieces into a global understanding $\mathcal{F}(A)$.

In the context of ASI systems:

- Discovery is not just trial and error, but coherent integration of partial knowledge.
- Surprise arises when gluing fails—when local pieces suggest incompatible global conclusions.

- Theory-building becomes a global section problem over the architecture space.

---

4. The Category of Presheaves: Epistemology of ASI Systems

Even before enforcing sheaf conditions, we can study the full **category of presheaves** on $\mathcal{C}_{\text{Arch}}$:

$$\widehat{\mathcal{C}_{\text{Arch}}} := \mathbf{Set}^{\mathcal{C}_{\text{Arch}}^{\text{op}}}$$

This is a topos—a rich categorical structure that behaves like a generalized universe of sets. In this topos:

- Presheaves represent possible knowledge states of the ASI system.
- Morphisms between presheaves encode information flow, such as refinement, correction, or generalization.
- Logical operations (e.g., conjunction, implication, quantification) can be interpreted internally within the topos, giving rise to a formal logic of discovery.

In this view, an ASI system is not just generating architectures—it is navigating a topos, evolving its internal logic and knowledge state as new observations accumulate.

Summary

By introducing a Grothendieck topology over the architecture space and interpreting observables as sheaves, we give a precise mathematical meaning to the idea of discovery as synthesis. Autonomous AI systems are no longer just performing black-box optimization—they are engaging in structured, logic-preserving exploration of a rich epistemic landscape. The resulting topos framework provides a global, interpretable foundation for scientific reasoning, extending far beyond neural architecture search and into the future of AI-guided knowledge creation itself.

VIII. Comparison with ASI-Arch and Other NAS Systems

The current generation of Neural Architecture Search (NAS) systems—including the recently unveiled ASI-Arch—have achieved impressive results, yet they remain grounded in a fundamentally heuristic and empirical paradigm. In contrast, our proposed categorical and homotopical framework shifts the foundation of architecture discovery from trial-based sampling to principled, structure-aware reasoning. Below, we offer a structured comparison that highlights the epistemological and operational differences between these approaches.

---

1. Heuristic Sampling vs. Structure-Preserving Transformations

ASI-Arch and conventional NAS operate by:

- Random or guided sampling from a search space defined by configurable parameters (e.g., layer types, connectivity patterns, hyperparameters).
- Exploring models by mutation, random walk, or black-box optimization strategies (e.g., evolutionary algorithms, reinforcement learning).

Such techniques lack a notion of structural integrity: changes may be arbitrary and often result in invalid or redundant configurations unless filtered post hoc.

By contrast, our framework:

- Defines transformations as morphisms in a category, ensuring they are semantically valid, compositional, and interpretable.
- Allows structure to evolve via diagrammatic reasoning, preserving functional integrity across transformations.
- Supports homotopic deformations to understand families of functionally equivalent models.

This enables not just exploration, but guided traversal of the architecture space—prioritizing *meaningful change* over random mutation.

---

2. Metrics-Based Reward vs. Categorical Inference

In ASI-Arch and most NAS systems:

- The discovery process is driven by scalar reward signals derived from model performance (e.g., accuracy, FLOPs, loss).
- Agents optimize their behavior through reinforcement or black-box gradient-free strategies.
- There is no deeper understanding of *why* one model is better than another—only that it *is*.

Our categorical model reframes evaluation as:

- A functorial map from the architecture category to an evaluation category, preserving structure between models and their empirical manifestations.
- Evaluation is no longer a reward—it is an inference grounded in internal logic, enabling:
    - Identification of equivalence classes
    - Detection of novelty via disconnected components
    - Tracking of epistemic dependencies (e.g., what prior models a new discovery builds upon)

This opens the door to explainable discovery, where outcomes are not just measured, but understood in terms of their generative pathways.

---

3. Interpretability, Reproducibility, and Generalization

Interpretability:

- ASI-Arch treats architectures as opaque strings or graphs; no formal system relates syntactic variation to semantic function.
- Our framework introduces commutative diagrams and homotopy types, enabling symbolic and visual interpretability.

Reproducibility:

- NAS results often depend on random seeds, fragile pipelines, and non-deterministic hardware.
- In our model, discoveries correspond to diagrammatic paths in a structured space. If a diagram commutes, its logic holds independently of stochastic variation.

Generalization:

- ASI-Arch and similar systems often specialize on narrow tasks (e.g., CIFAR, ImageNet), risking overfitting to the search environment.
- Our approach models discovery across a topos of knowledge, enabling reuse of structural motifs, evaluation sheaves, and reasoning pathways across tasks.

In essence, where NAS searches *blindly*, our categorical framework reasons with awareness—not only of the space being explored, but of the relationships that bind it together.

Summary Table

Aspect	ASI-Arch / NAS	Categorical-Homotopical Framework
Search Method	Heuristic sampling	Structure-preserving morphisms
Evaluation	Scalar reward	Functorial inference
Model Identity	Structural hash / string	Homotopy equivalence
Novelty Definition	Delta in performance	New homotopy class or disconnected component
Learning Process	Reinforcement or evolution	Higher-order functor adaptation
Epistemic Integration	Absent	Sheaf-theoretic gluing
Result Type	Empirical	Mathematical theorem with constructible proof
Reproducibility	Limited	Guaranteed by diagram commutativity

By grounding architecture discovery in the mathematics of categories, functors, homotopies, and sheaves, we provide not just a toolkit for better models, but a new epistemology for AI itself—a logic for machine-led science.

IX. Conclusion

In this paper, we have presented a principled mathematical framework for understanding and guiding autonomous architecture discovery, using the tools of category theory, homotopy theory, and topos theory. By formalizing neural architectures as objects in a category, structural edits as morphisms, and functional equivalence as homotopies, we have created a rigorous language in which to speak of novelty, discovery, and transformation. We then extended this

framework to higher-order structures—modeling autonomous agents as functors, their interactions as 2-morphisms, and their epistemic integration as gluing in a sheaf over a Grothendieck topology. The result is a deeply interconnected mathematical ecosystem where architecture search is no longer a black-box optimization problem, but a coherent traversal of a richly structured landscape.

This shift is not merely academic. As AI systems become more autonomous, their operations—and the knowledge they generate—will increasingly transcend human intuition. To keep pace, we need emergent mathematics for emergent systems. Heuristic tuning and scalar metrics can no longer explain what these systems are doing. We must instead develop models that mirror the compositional, recursive, and epistemic logic of autonomous AI itself.

Future Directions

Several fertile directions arise from this formalization:

- AI-Based Proof Systems: With architecture discovery cast in categorical and logical terms, it becomes possible to build AI agents that not only generate models but prove theorems about their structure, behavior, and optimality. These agents would not merely search—they would reason.
- Categorical Scientific Agents: The notion of an autonomous scientific agent can be recast as a bundle of interacting functors and sheaves, evolving over time via higher-order transformations. These agents could be trained not just to discover patterns in data, but to build global theories consistent with observed phenomena.
- Functorial Learning Pipelines: Entire machine learning workflows—including preprocessing, modeling, evaluation, and interpretation—can be modeled as functors between categories. This opens the door to machine-level abstraction, where workflows can be composed, reasoned about, and transformed with mathematical guarantees.
- Topos-Theoretic Knowledge Integration: By situating discovery within a topos of presheaves, we can develop AI systems capable of reconciling contradictory observations, synthesizing local knowledge into global

understanding, and generating new types within a self-evolving logical universe.

In short, our goal is to transition from empirical machines to epistemic machines—entities that do not merely compute, but understand. The categorical and homotopical tools outlined in this paper offer a foundation for this transformation, marking a path toward an AI that is not just intelligent, but scientifically coherent.

Appendices

Appendix A. Categorical Definitions and Diagrams

To support the formalism developed in the main text, we summarize core categorical constructs used throughout the paper and illustrate them with relevant diagrams.

A1. Definitions

- **Category** $\mathcal{C}$: A collection of **objects** and **morphisms** (arrows) satisfying:
  - Identity: $\forall A \in \mathrm{Ob}(\mathcal{C}), \exists\, \mathrm{id}_A : A \to A$
  - Composition: $f : A \to B, g : B \to C \Rightarrow g \circ f : A \to C$
  - Associativity: $h \circ (g \circ f) = (h \circ g) \circ f$
- **Functor** $F : \mathcal{C} \to \mathcal{D}$: A mapping of objects and morphisms that preserves identities and composition:

$$F(\mathrm{id}_A) = \mathrm{id}_{F(A)}, \quad F(g \circ f) = F(g) \circ F(f)$$

- **Natural Transformation** $\eta : F \Rightarrow G$: A family of morphisms $\eta_A : F(A) \to G(A)$ such that for every morphism $f : A \to B$,

$$\eta_B \circ F(f) = G(f) \circ \eta_A$$

  (This forms a **commuting square**.)

- **Site**: A category equipped with a **Grothendieck topology**, which assigns to each object a collection of covering families.
- **Sheaf**: A presheaf satisfying a gluing condition over coverings—compatible local data can be uniquely "glued" to a global section.

A2. Diagrams

Diagram 1: Functorial Mapping

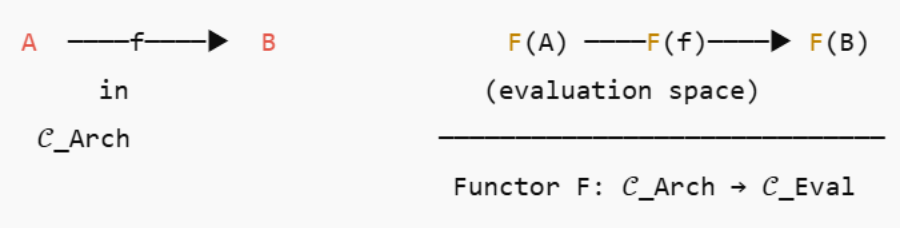
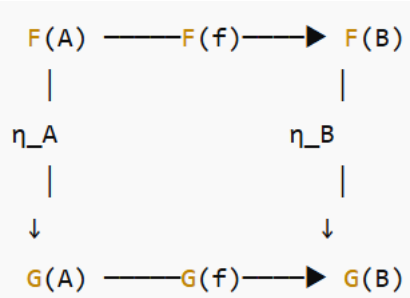


Diagram 2: Commuting Square (Natural Transformation)



Appendix B. Illustrative Homotopies Between Architecture Classes

Homotopies in the architecture space capture transformations that preserve functional behavior. Below are conceptual examples.

B1. Homotopy: Dense → Residual Block

> Let A be a densely connected feedforward network and B be a residual network. A homotopy $H : A \simeq B$ might involve:

1. Add identity paths between layers (preserving gradient flow).
2. Replace layer-by-layer computation with residual maps.
3. Each intermediate step yields a functionally equivalent output.

Diagrammatically:

```
A = DenseNet0  → DenseNet1 → ... → ResNetN = B
```

B2. Homotopy Class within Transformer Variants

Let *A* and *B* be two Transformer architectures differing in:

- Attention mechanism (softmax vs. linear)
- Positional encoding

If their outputs are empirically indistinguishable and their representations collapse to the same functional behavior across tasks, they are homotopic under our evaluation functor *F*.

Appendix C. Glossary of HoTT and Category Theory Concepts

Term	Definition
Type	A collection of terms (like a set); interpreted as a space in HoTT
Path	A homotopy between terms, treated as a witness of equality
Univalence Axiom	Equivalence between types implies identity: $A \simeq B \Rightarrow A = B$
∞-Groupoid	A structure where objects, morphisms, 2-morphisms, etc., all exist and compose
Presheaf	A contravariant functor from a category $\mathcal{C}$ to $\mathbf{Set}$
Sheaf	A presheaf satisfying compatibility and gluing over a Grothendieck topology
Topos	A category resembling $\mathbf{Set}$, supporting internal logic, limits, and colimits

---

Appendix D. Mapping NAS Search Spaces to Categorical Spaces

Most NAS systems define search spaces in terms of parameterized templates. We reinterpret these as categorical constructs.

NAS Element	Categorical Analogue
Architecture template (e.g., CNN cell)	Object in $\mathcal{C}_{\text{Arch}}$
Mutation (add/remove layer)	Morphism in $\mathcal{C}_{\text{Arch}}$
Search path	Functorial traversal or path in morphism space
Architecture equivalence	Homotopy or natural transformation
Reward-based policy update	Morphism in $\mathcal{C}_{\text{Eval}}$, possibly lacking functorial structure
Graph of sampled architectures	Underlying 1-skeleton of the category
Meta-NAS (NAS over NAS)	Higher-category with morphisms between functors (2-morphisms)

This mapping allows NAS processes to be reinterpreted as diagrammatic journeys through structured space, replacing sampling with informed traversal, and treating discovery as coherent morphogenesis rather than blind optimization.

These appendices provide the structural scaffolding for future implementation of mathematically-aware autonomous AI researchers—agents that don't just search for better architectures but inhabit a structured, intelligible world of scientific meaning.

References

1. Baez, J., & Stay, M. (2011). *Physics, topology, logic and computation: a Rosetta Stone*. In New Structures for Physics (pp. 95–172). Springer. https://doi.org/10.1007/978-3-642-12821-9_2
2. Awodey, S. (2010). *Category Theory* (2nd ed.). Oxford University Press.
3. Riehl, E. (2016). *Category Theory in Context*. Dover Publications.
4. Lawvere, F. W., & Schanuel, S. H. (2009). *Conceptual Mathematics: A First Introduction to Categories* (2nd ed.). Cambridge University Press.
5. Lurie, J. (2009). *Higher Topos Theory*. Princeton University Press.
6. Univalent Foundations Program. (2013). *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study. https://homotopytypetheory.org/book/
7. Shulman, M. (2019). *Homotopy type theory: the logic of space*. In *The Stanford Encyclopedia of Philosophy* (Fall 2019 Edition), Edward N. Zalta (ed.). https://plato.stanford.edu/entries/homotopy-type-theory/
8. Mac Lane, S. (1998). *Categories for the Working Mathematician* (2nd ed.). Springer.
9. Spivak, D. I. (2014). *Category Theory for the Sciences*. MIT Press.
10. Voevodsky, V. (2014). *A simple type system with two identity types*. https://www.math.ias.edu/vladimir/files/2014_HoTT_identity_types.pdf
11. Elsken, T., Metzen, J. H., & Hutter, F. (2019). *Neural Architecture Search: A Survey*. Journal of Machine Learning Research, 20(55), 1–21.
12. Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2019). *Regularized Evolution for Image Classifier Architecture Search*. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(1), 4780–4789.
13. Liu, H., Simonyan, K., & Yang, Y. (2019). *DARTS: Differentiable Architecture Search*. In *International Conference on Learning Representations*.

14. Zoph, B., & Le, Q. V. (2017). *Neural Architecture Search with Reinforcement Learning*. In *International Conference on Learning Representations*.

15. Liu, H., Simonyan, K., Vinyals, O., Fernando, C., & Kavukcuoglu, K. (2018). *Hierarchical Representations for Efficient Architecture Search*. In *International Conference on Learning Representations*.

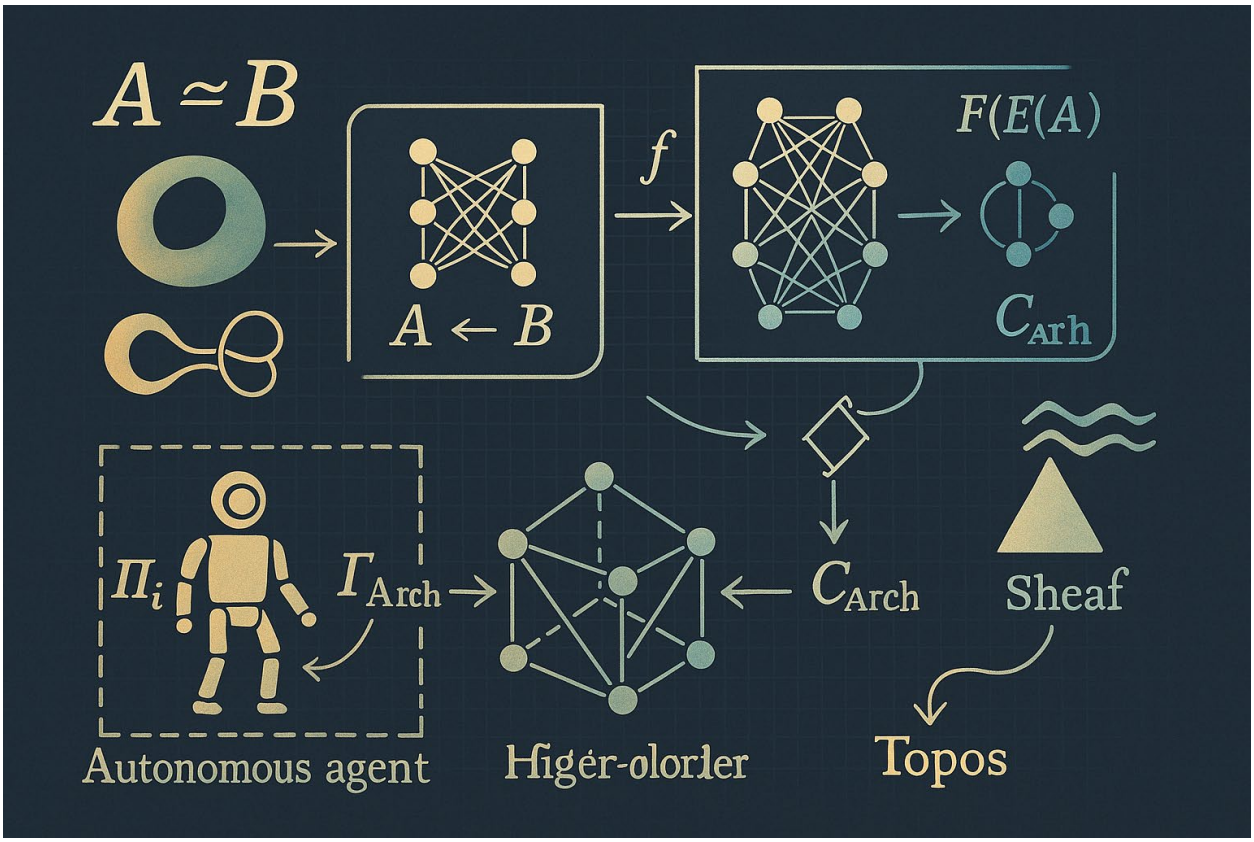
16. Gao, B., Shi, H., Xie, M., et al. (2025). *AlphaGo Moment for Model Architecture Discovery*. arXiv:2507.18074 [cs.LG]. https://arxiv.org/abs/2507.18074

17. Coecke, B., Kissinger, A. (2017). *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press.

18. Spivak, D. I., & Kent, R. E. (2012). *Ologs: A categorical framework for knowledge representation*. PLoS ONE, 7(1), e24274. https://doi.org/10.1371/journal.pone.0024274

19. Fong, B., & Spivak, D. I. (2019). *Seven Sketches in Compositionality: An Invitation to Applied Category Theory*. Cambridge University Press.

20. Abramsky, S., & Coecke, B. (2004). *A categorical semantics of quantum protocols*. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science*, 415–425.


$A \simeq B$
$A \leftarrow B$
f
$F(E(A)$
C_{Arch}
Π_i
Γ_{Arch}
C_{Arch}
Sheaf
Autonomous agent
Higher-order
Topos